31.5 Exerises

Factual

1. For each of the following, state whether they prevent, exacerbate, or have no effect on information leakage.
   - using unnecessary print statements to show the state of the program while it's running
   - employing temporal decomposition when architecting your code
   - creating deep modules
   - using helper methods and classes
2. In Woolley's study, which factors correlated with the quality of a team's output?

Problem 1

The key to remember is that information leakage is *not* related to the state of the program. Instead, it's when multiple pieces of code reflect a single design decision/module, introducing unnecessary complexity and dependencies.

- **Using unnecessary print statements to show the state of the program while it's running:** no effect on information leakage.
- **Employing temporal decomposition when architecting your code**: exacerbates information leakage.
- **Creating deep modules**: prevents information leakage.
- **Using helper methods and classes**: prevents information leakage.

Problem 2

The two most important factors were *turn-taking* during conversations and the average ability of the group members to *recognize emotional state* from a

person's eyes.

Metacognitive

1. Give some examples of unnecessary obscurity that contributes to complexity.
2. Give some examples of unnnecessary dependencies that contribute to complexity.

Problem 1

There are many kinds of obscurity that can contribute to complexity. These include putting too much code into one function or module, having a large amount of variables that need to be maniuplated, and not breaking up your code or having helper functions.

Problem 2

One example is copy-pasting code across different modules. This means that to change this block of code, you have to change every location that this code appears in.

Previous
31.4 Teamwork

Next
32. More Quick Sort, Sorting Summary

Last updated 1 year ago